

報告番号	※甲	第	号
------	----	---	---

## 主 論 文 の 要 旨

論文題目 組込みシステムに適したコンポーネントシステム

氏 名 安積 卓也

## 論 文 内 容 の 要 旨

近年、組込みシステムの複雑化・大規模化に伴いソフトウェア生産性が問題となっている。ソフトウェア生産性を向上させる手法として、汎用システムでは、ソフトウェアを機能ごとに分割し得られたソフトウェア部品(コンポーネント)を組み合わせるによりソフトウェアを開発する技術が広く使用されている。これをコンポーネント技術と呼ぶ。また、コンポーネント技術を実現するためのシステムをコンポーネントシステムと呼ぶ。汎用システムにおけるコンポーネントシステムとして、JavaBeans, COM, CCM (CORBA Component Model) などがある。一方、組込みシステムでも、特定の分野でのコンポーネントシステム (PBO, Koala, AUTOSAR, SAVE CCT など) が注目されつつある。

組込みシステムでは、メモリなどのリソース制約が厳しく、さらにリアルタイム性も求められる。そのため、コンポーネントシステムを用いることによるメモリ使用量や処理時間の増大を、できる限り避けなければならない。汎用システムでのコンポーネント技術では、実行時に動的にコンポーネントのインスタンス化や結合(以下、動的結合と呼ぶ)が行われる。動的結合では、インスタンス化や結合にかかるオーバーヘッドが大きい。

上記の問題を解決するために、組込みシステムに適したオーバーヘッドの小さいコンポーネントシステムとして、TOPPERS Embedded Component System (TECS) を提案する。本コンポーネントシステムは、コンポーネントを静的にインスタンス化・結合する静的なコンポーネントモデルを採用する。コンポーネントの属性情報やコンポーネント間を結合するためのコードはインタフェースジェネレータというツールにより静的に生成されるため、インスタンス化や結合にかかる実行時のオーバーヘッドをなくせることに加えて、メモリの使用量を削減できるという利点もある。さらに、コンポーネント間を結合するコードをインタフェースジェネレータ

により最適化することで、コンポーネントを呼び出すオーバーヘッドを最小限に抑えることができる。本論文では、シリアル通信インタフェースドライバのコンポーネント化を例として、コンポーネント化に伴うオーバーヘッドやソフトウェアの記述量などを評価することで、本コンポーネントシステムの有用性を示す。

次に、コンポーネントベース開発におけるソフトウェアの生産性の向上を目的とし、コンポーネントの結合部分に影響を与えずにコンポーネントを挿入する機構を提案する。コンポーネントの結合部分に新たなコンポーネントを挿入することで、さまざまな応用が可能である。たとえば、トレースログ機構、アクセス制御機構、RPCチャネル、性能評価、排他制御機構など広範囲に適用できる。提案機構はインタフェースジェネレータのプラグインとして実装される。そのプラグインを選択することで、挿入されるコンポーネントはコンポーネントの結合に適したコンポーネント記述、およびC言語のソースコードが自動生成され利用される。TECSは、他のコンポーネントシステムと比べ、セマフォなど粒度が小さいものまでコンポーネントとして扱える。そのため、さまざまな箇所にコンポーネントを挿入できる。

さらに、 $\mu$ ITRON4.0仕様やOSEK/VDX仕様などのリアルタイムOS上で資源を保護することを目的とし、組込みシステム向けコンポーネントシステムを利用したアクセス制御機構を提案する。ソフトウェアの大規模化に伴うバグによる障害への対処、サードパーティのコンポーネントの安全な利用、ネットワーク接続に伴う外部からの不正アクセス防止の理由から、組込みシステムにおいても、資源を保護するためのアクセス制御が必要になってきた。しかし、リアルタイムOSが管理しているハードウェア資源は、プロセッサ、メモリ、タイマのみであり、ソフトウェア資源もタスクなどに限られている。そのため、ファイルなどの資源をOSで管理している汎用システムのように、OS内部でアクセス制御を行っても十分ではない。そこで、本研究では、アクセス制御対象の資源をコンポーネントが管理するものとし、コンポーネントに対するアクセス制御機構の導入方法を提案する。さらに、アクセス制御機構の導入によるオーバーヘッドを評価する。

最後に、さまざまなコンポーネントの呼出し形態で、コンポーネントを再利用することを目的とし、メモリ共有を考慮したRPCシステムを提案する。呼出し形態は(1)同タスク同プロセッサ、(2)別タスク同プロセッサ、(3)メモリを共有する別プロセッサ、(4)メモリを共有しない別プロセッサ(ネットワーク接続など)の四つに分類される。組込みシステムでは、特に(2)と(3)では汎用システムとは異なり、メモリの種類・用途を考慮する必要がある。想定される通信チャネルも多様である。したがって(2)と(3)の呼出し形態に対して、既存のRPCシステムをそのまま適用することは容易ではない。さらにメモリ領域の確保・解放を明記するためのインタフェース記述を提案する。従来インタフェース記述と比較することで、提案インタフェース記述の有用性を示す。